

13.08.19

# Digital Image Processing (CSE/ECE 478)

Lecture 5 : Using Histogram Statistics for Image Enhancement, Introduction to Spatial Filters



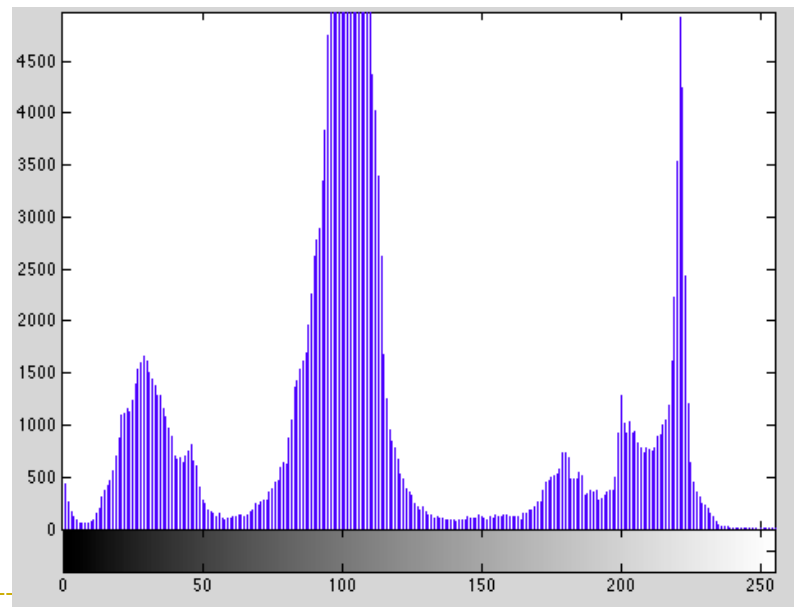
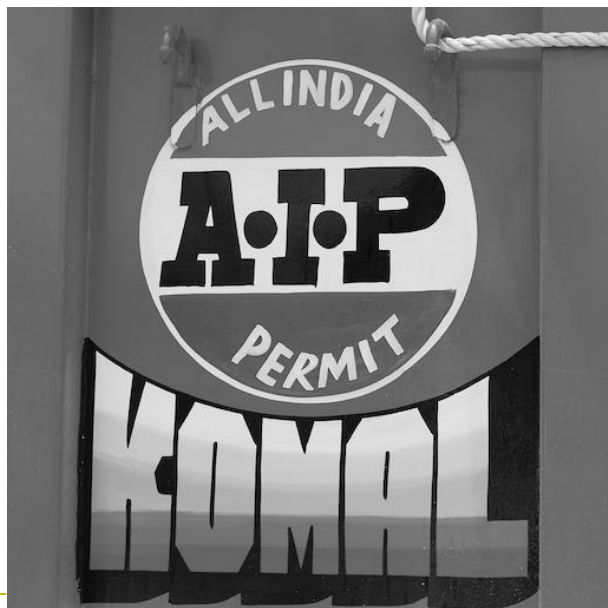
Ravi Kiran

# Histogram

$$h_r(i) = n_i$$

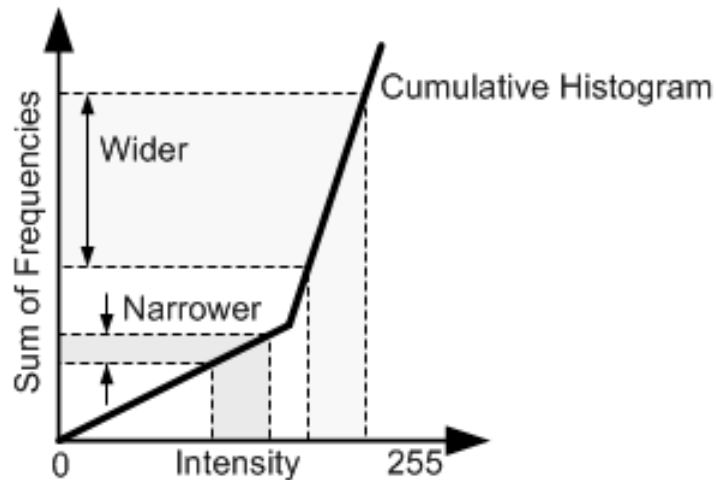
$i \rightarrow$  intensity value, range  $[0 L-1]$

$n_i \rightarrow$  number of pixels with intensity  $i$



# Histogram Equalization

---

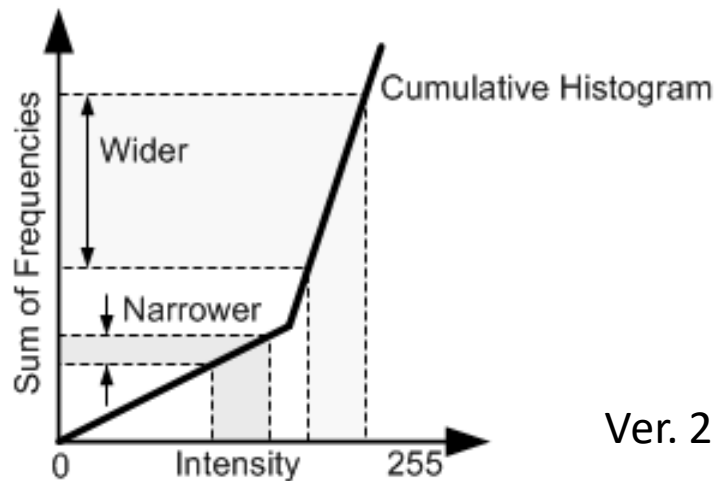


$$s = T(r) = (L - 1) \int_0^r p_r(w) dw$$

$$s_k = T(r_k) = \text{round} \left( (L - 1) \sum_{j=0}^{j=k} p_r(r_j) \right)$$

# Histogram Equalization

$$h[i] = \text{constant}, \quad 0 \leq i \leq L - 1$$



$$s = T(r) = (L - 1) \int_0^r p_r(w) dw$$

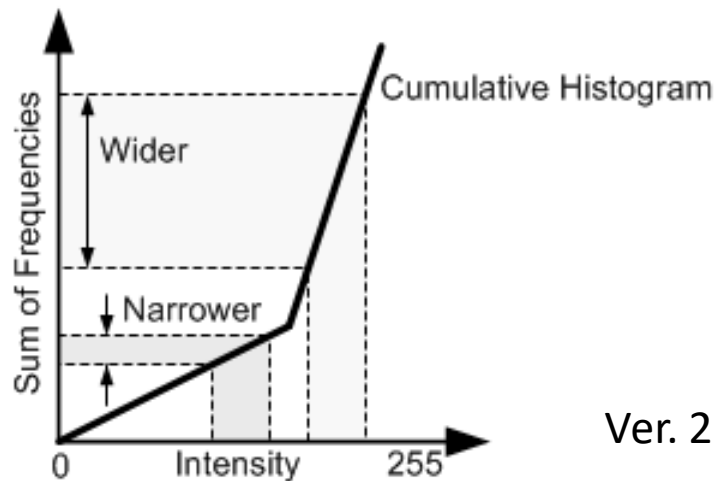
$$s_k = T(r_k) = \text{round} \left( (L - 1) \underbrace{\sum_{j=0}^{j=k} p_r(r_j)} \right)$$

Ver. 2

$$s_k = T(r_k) = \text{round} \left( (L - 1) * \frac{\text{cdf}(r_k) - \text{cdf}_{\min}}{1 - \text{cdf}_{\min}} \right)$$

# Histogram Equalization

$$h[i] = \text{constant}, \quad 0 \leq i \leq L - 1$$



$$s = T(r) = (L - 1) \int_0^r p_r(w) dw$$

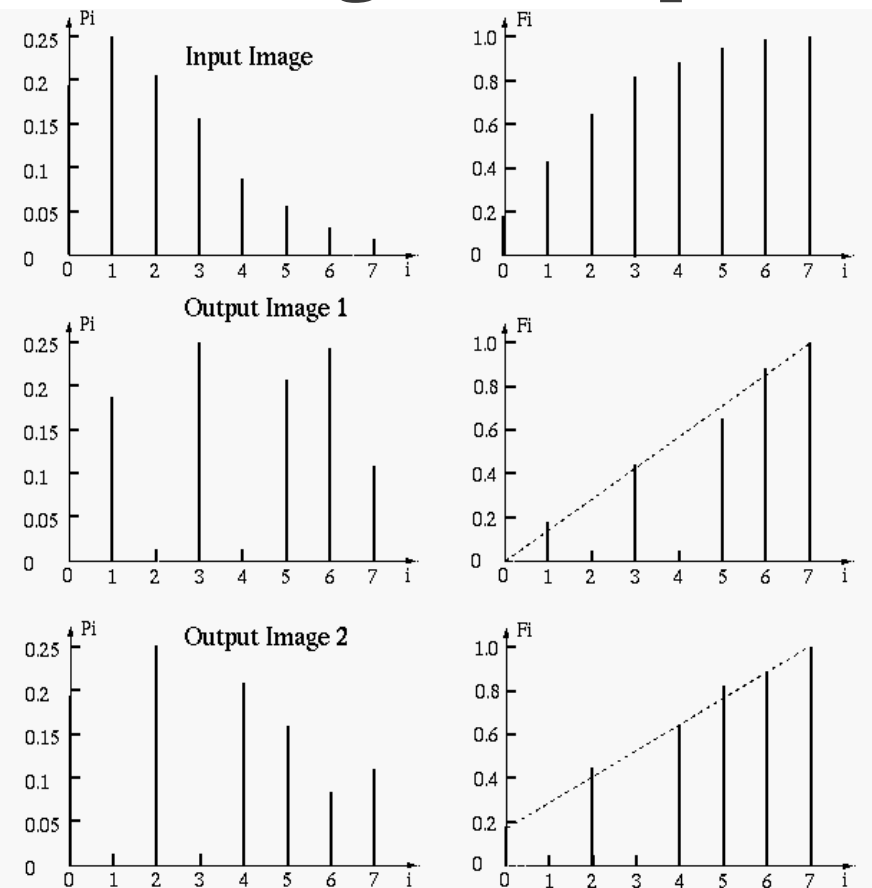
$$s_k = T(r_k) = \text{round} \left( (L - 1) \underbrace{\sum_{j=0}^{j=k} p_r(r_j)} \right)$$

Ver. 2 
$$s_k = T(r_k) = \text{round} \left( (L - 1) * \frac{cdf(r_k) - cdf_{min}}{1 - cdf_{min}} \right)$$

$$cdf_{min} = p_r(r_a) \text{ where } r_a = \min\{r_t | p_r(r_t) > 0\}; 0 \leq r_t \leq (L - 1)$$



# Histogram Equalization



$$s_k = T(r_k) = \text{round} \left( (L - 1) \underbrace{\sum_{j=0}^{j=k} p_r(r_j)} \right)$$

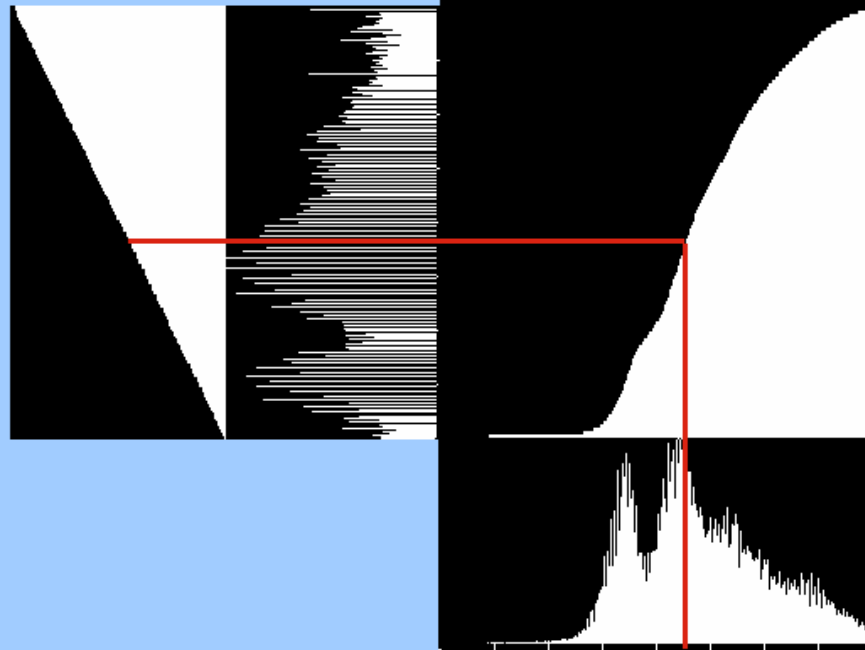
**Ver. 2**  $s_k = T(r_k) = \text{round} \left( (L - 1) * \frac{cdf(r_k) - cdf_{min}}{1 - cdf_{min}} \right)$

$$cdf_{min} = p_r(r_a) \text{ where } r_a = \min\{r_t | p_r(r_t) > 0\}; 0 \leq r_t \leq (L - 1)$$

# Histogram Equalization



Equalized histogram



Histogram of original image



# Histogram Processing

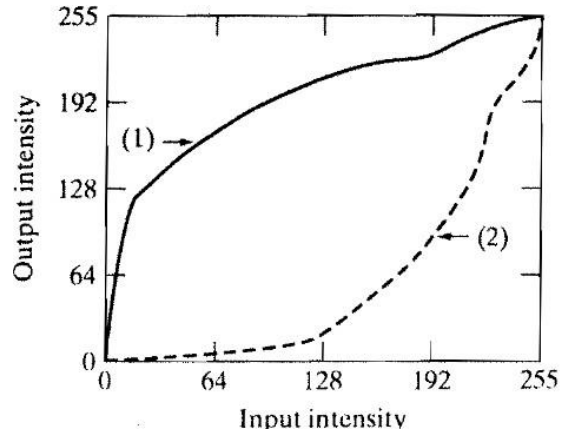
---

- ▶ Histogram Equalization
- ▶ Histogram Specification
- ▶ Local Histogram Equalization





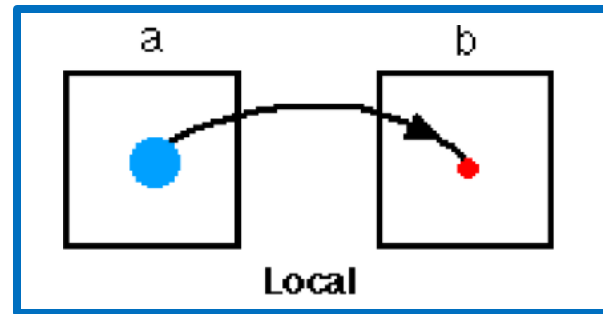
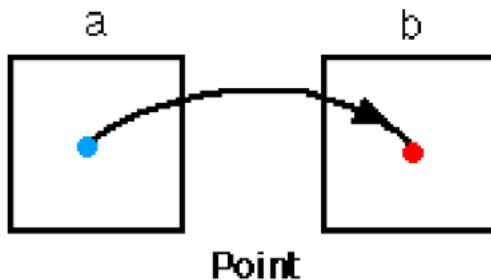
# Histogram Specification / Matching



---

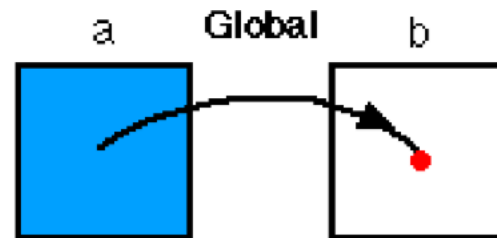
- ▶ Manipulating Pixels Directly in Spatial Domain

- ▶ Point to Point



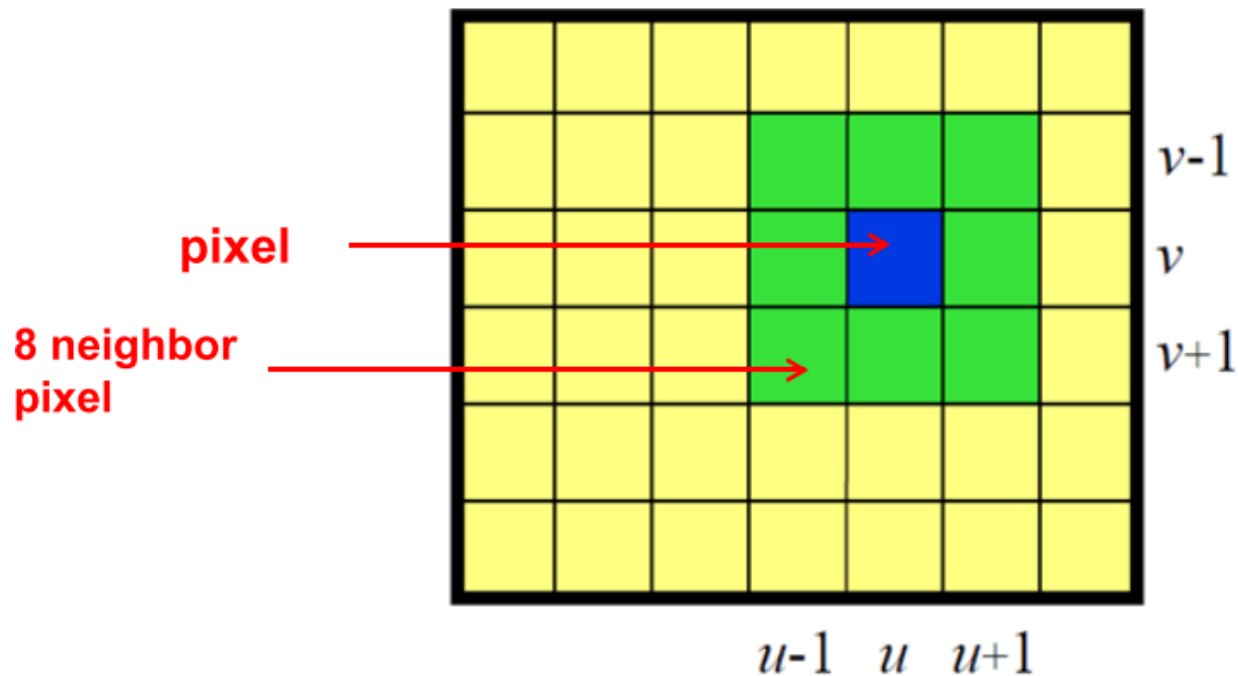
- ▶ **Neighborhood to Point**

- ▶ Global Attribute to Point



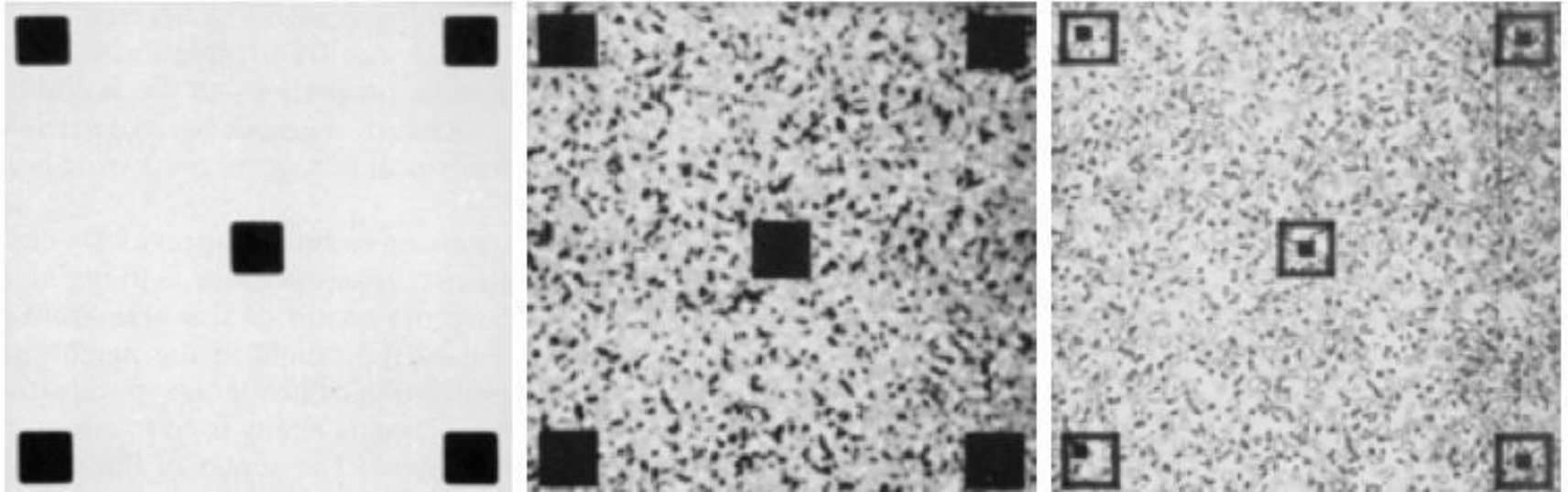
# Neighborhood

---



# Local Histogram Processing

---



# Using Histogram Statistics for Image Enhancement

---

- we use some statistical parameters

- global:

- $p(r_i) = \frac{n_i}{n}$

- $m(r) = \sum_{i=0}^{L-1} p(r_i) r_i$

- $\sigma^2(r) = \sum_{i=0}^{L-1} (r_i - m)^2 p(r_i)$

- local:

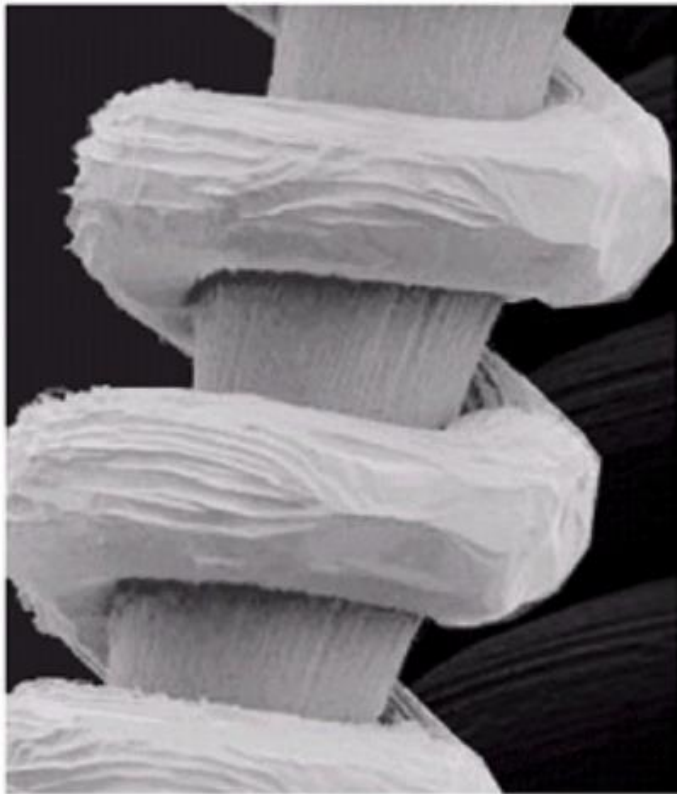
- $p(r_{s,t})$ : neighborhood normalized histogram at coordinates  $(s, t)$  using a mask centered at  $(x, y)$

- $m_{S_{xy}} = \sum_{(s,t) \in S_{xy}} p(r_{s,t}) r_{s,t}$

- $\sigma^2(S_{xy}) = \sum_{(s,t) \in S_{xy}} [r_{s,t} - m_{S_{xy}}]^2 p(r_{s,t})$



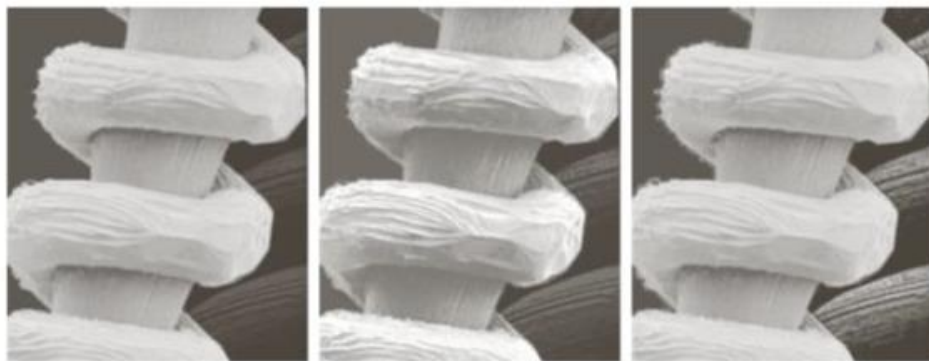
# Using Histogram Statistics for Image Enhancement



- we use some statistical parameters
  - global:
    - $p(r_i) = \frac{n_i}{n}$
    - $m(r) = \sum_{i=0}^{L-1} p(r_i) r_i$
    - $\sigma^2(r) = \sum_{i=0}^{L-1} (r_i - m)^2 p(r_i)$
  - local:
    - $p(r_{s,t})$ : neighborhood normalized histogram at coordinates  $(s, t)$  using a mask centered at  $(x, y)$
    - $m_{S_{xy}} = \sum_{(s,t) \in S_{xy}} p(r_{s,t}) r_{s,t}$
    - $\sigma^2(S_{xy}) = \sum_{(s,t) \in S_{xy}} [r_{s,t} - m_{S_{xy}}]^2 p(r_{s,t})$
- Objective: Enhance dark areas while leaving light areas unchanged
- Can we use local statistic to obtain it ?



# Using Histogram Statistics for Image Enhancement



a b c

**FIGURE 3.27** (a) SEM image of a tungsten filament magnified approximately 130 $\times$ . (b) Result of global histogram equalization. (c) Image enhanced using local histogram statistics. (Original image courtesy of Mr. Michael Shaffer, Department of Geological Sciences, University of Oregon, Eugene.)

$$g(x, y) = \begin{cases} E[f(x, y)], & \text{if } m_s \leq k_0 m_G \text{ and } k_1 \sigma_G \leq \sigma_s \leq k_2 \sigma_G \\ f(x, y), & \text{otherwise} \end{cases}$$

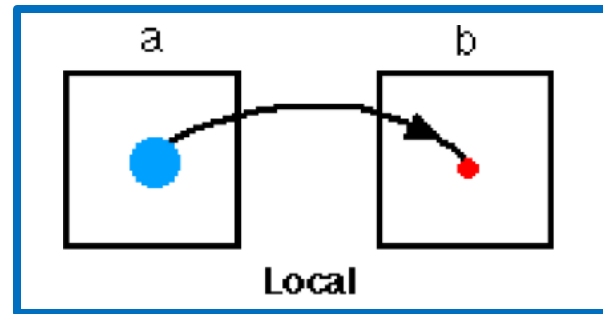
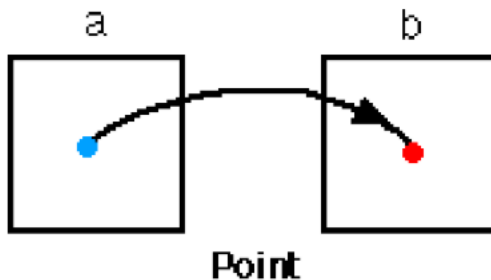
$m_G$  : global mean;     $\sigma_G$  : global standard deviation

$k_0 = 0.4$ ;     $k_1 = 0.02$ ;     $k_2 = 0.4$ ;     $E = 4$

---

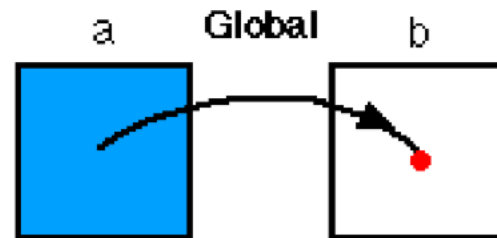
- ▶ Manipulating Pixels Directly in Spatial Domain

- ▶ Point to Point



- ▶ **Neighborhood to Point**

- ▶ Global Attribute to Point





# Spatial Domain Filtering

---

- ▶ What is a filter?
- ▶ What is filtering operation?



# Spatial Domain Filtering - Approaches

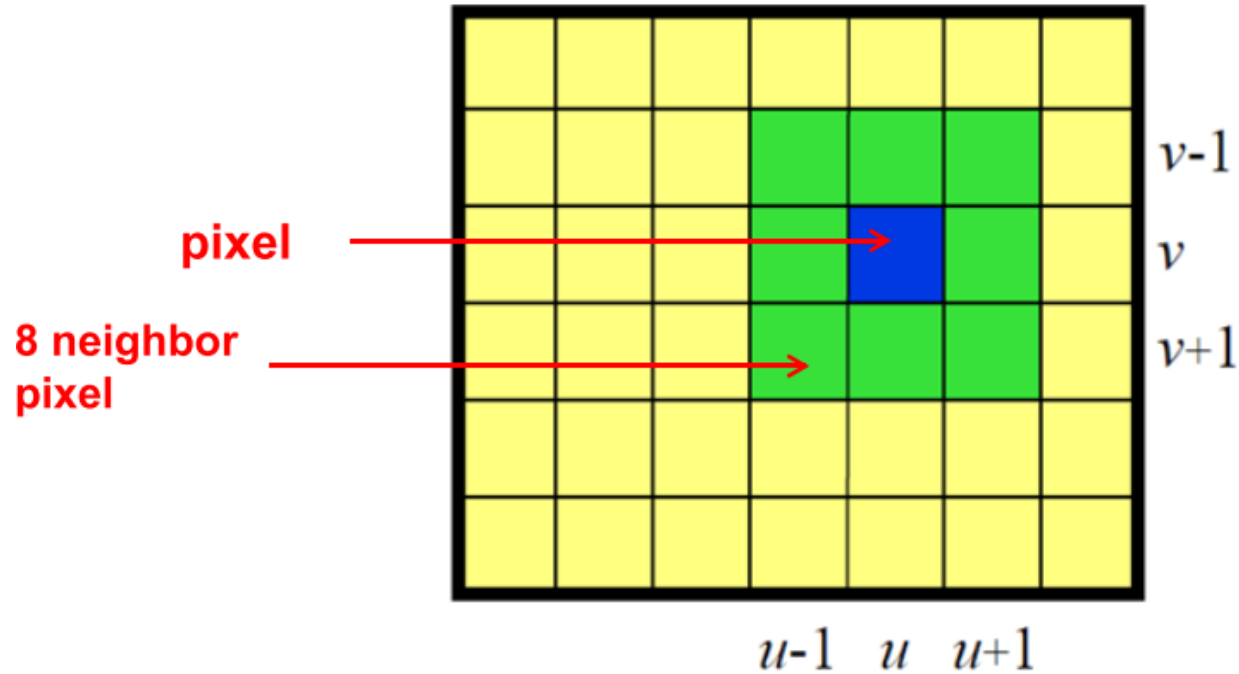
---

- ▶ Linear
- ▶ Non-linear

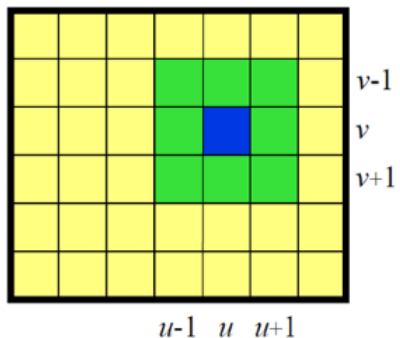


# Neighborhood Operation

---

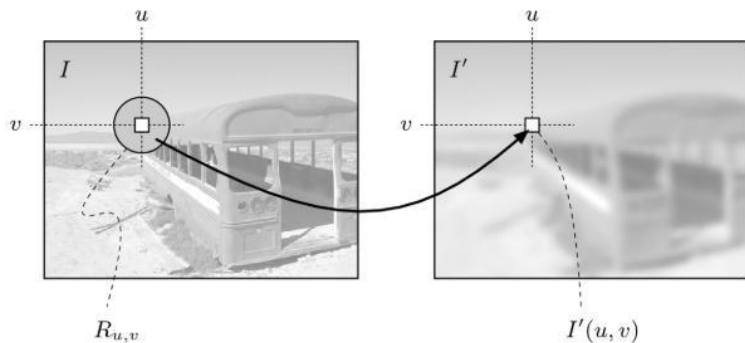


# Smoothing Operation



$$I'(u, v) \leftarrow \frac{p_0 + p_1 + p_2 + p_3 + p_4 + p_5 + p_6 + p_7 + p_8}{9}$$

$$I'(u, v) \leftarrow \frac{1}{9} \cdot [ I(u-1, v-1) + I(u, v-1) + I(u+1, v-1) + \\ I(u-1, v) + I(u, v) + I(u+1, v) + \\ I(u-1, v+1) + I(u, v+1) + I(u+1, v+1) ]$$



$$I'(u, v) \leftarrow \frac{1}{9} \cdot \sum_{j=-1}^1 \sum_{i=-1}^1 I(u+i, v+j)$$

# Smoothing as Averaging

I


H

→ Weight Mask  
Kernel  
Filter

$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$
$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$
$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$

$$I'(u, v) \leftarrow \frac{1}{9} \cdot \sum_{j=-1}^1 \sum_{i=-1}^1 I(u+i, v+j)$$

$$I'(u, v) \leftarrow \sum_{j=-1}^1 \sum_{i=-1}^1 I(u+i, v+j) \cdot H(i, j)$$

# Effect of Mask Size

---

Original Image



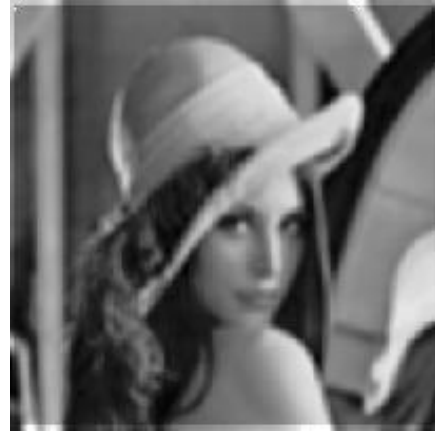
[3x3]



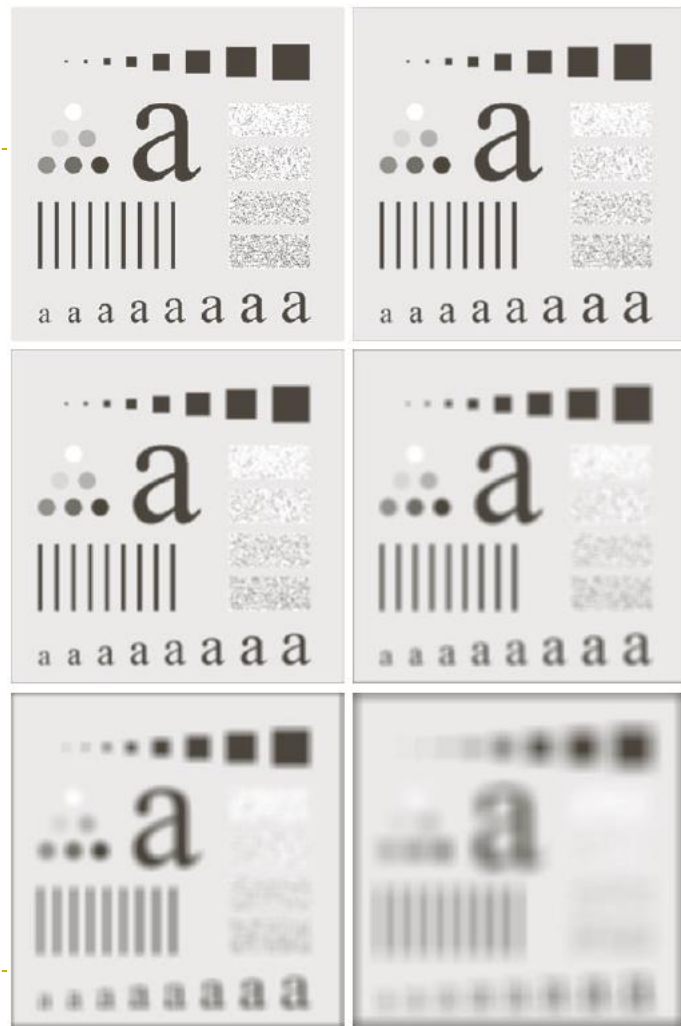
[5x5]



[7x7]



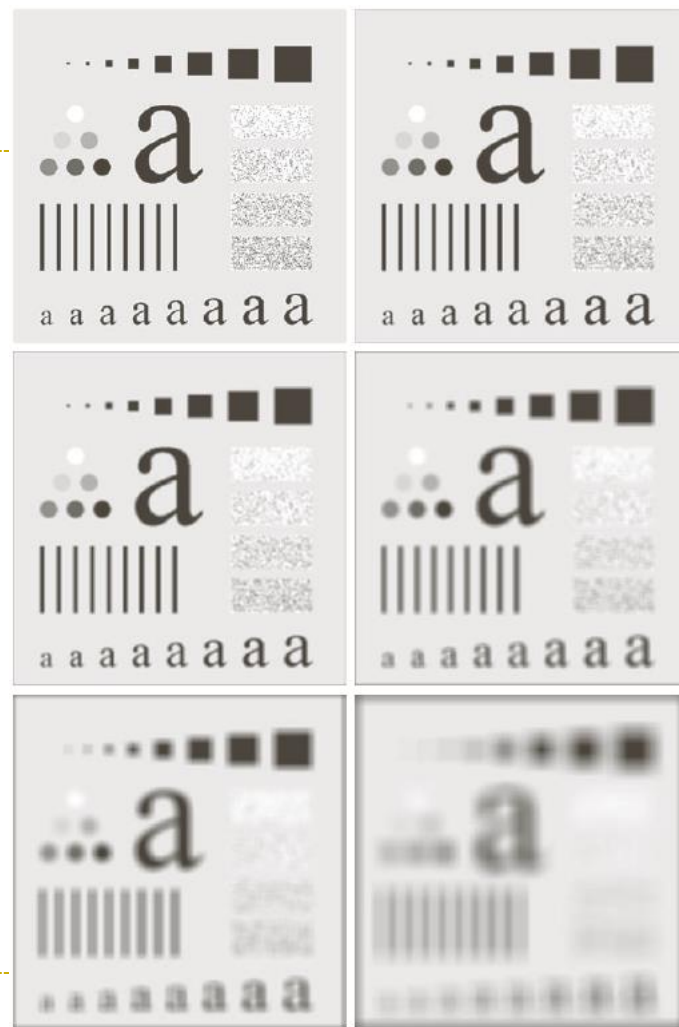
# Square averaging filter



# Square averaging filter

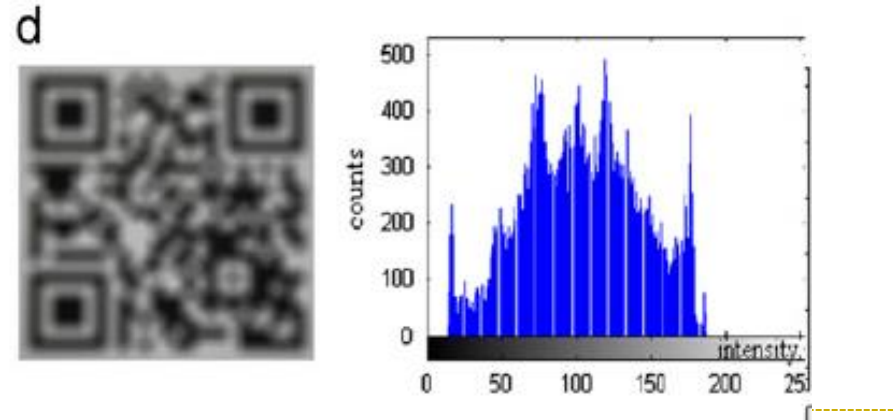
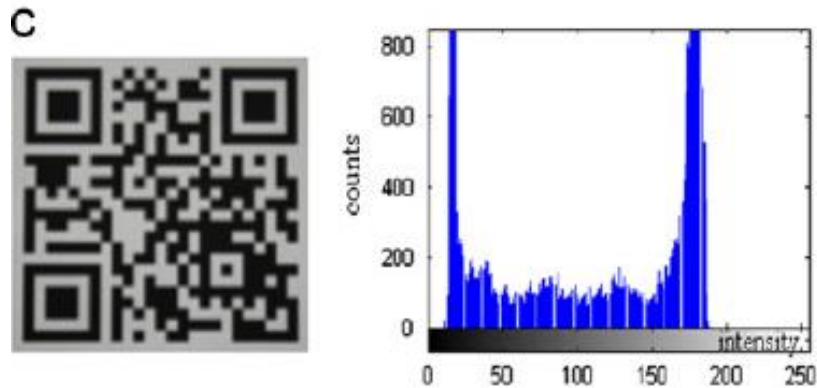
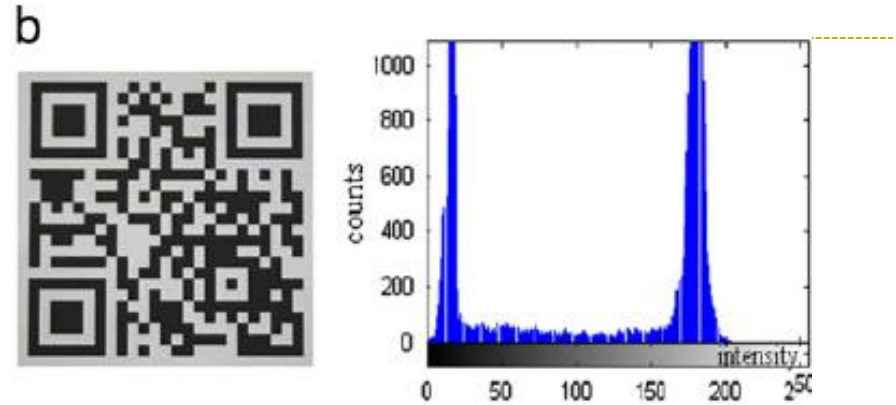
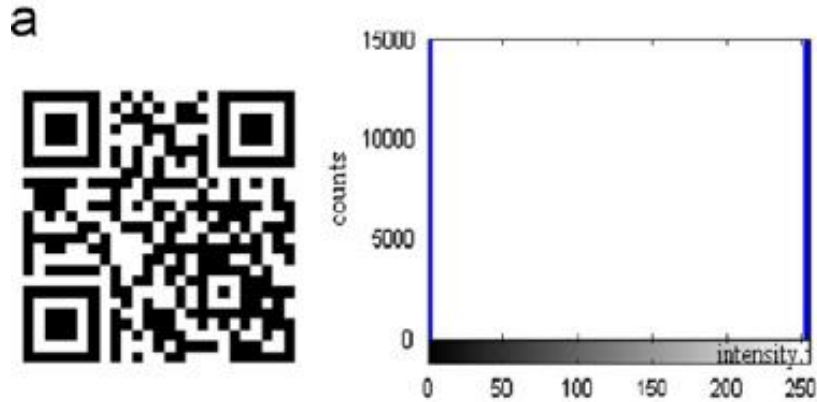
**FIGURE 3.33** (a) Original image, of size  $500 \times 500$  pixels. (b)–(f) Results of smoothing with square averaging filter masks of sizes  $m = 3, 5, 9, 15$ , and  $35$ , respectively. The black squares at the top are of sizes  $3, 5, 9, 15, 25, 35, 45$ , and  $55$  pixels, respectively; their borders are  $25$  pixels apart. The letters at the bottom range in size from  $10$  to  $24$  points, in increments of  $2$  points; the large letter at the top is  $60$  points. The vertical bars are  $5$  pixels wide and  $100$  pixels high; their separation is  $20$  pixels. The diameter of the circles is  $25$  pixels, and their borders are  $15$  pixels apart; their intensity levels range from  $0\%$  to  $100\%$  black in increments of  $20\%$ . The background of the image is  $10\%$  black. The noisy rectangles are of size  $50 \times 120$  pixels.

a b  
c d  
e f





# Smoothing – a histogram perspective



# Effect of Repeated Smoothing

---



Before



After



After repeated  
averaging

NOTE: Can get the effect of larger filters by  
smoothing repeatedly with smaller filters

---



# Weighted Averaging

$$I'(u, v) = \frac{\sum_{(j=-a)}^a \sum_{(i=-b)}^b I(u+i, v+j) \cdot H(i, j)}{\sum_{(j=-a)}^a \sum_{(i=-b)}^b H(i, j)}$$

 $\frac{1}{9} \times$ 

1	1	1
1	1	1
1	1	1

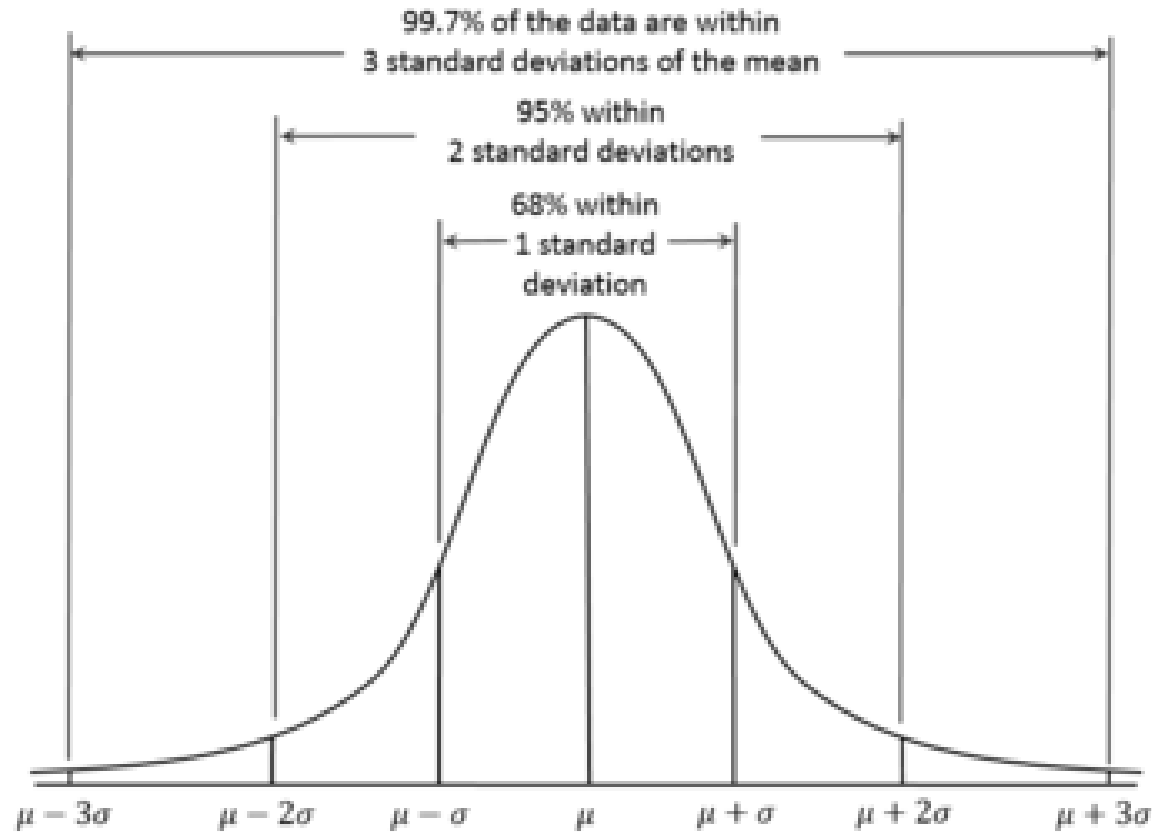
Standard average

 $\frac{1}{16} \times$ 

1	2	1
2	4	2
1	2	1

Weighted average

# Gaussian Function



$$y = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x - \mu)^2}{2\sigma^2}}$$

$\mu$  = Mean

$\sigma$  = Standard Deviation

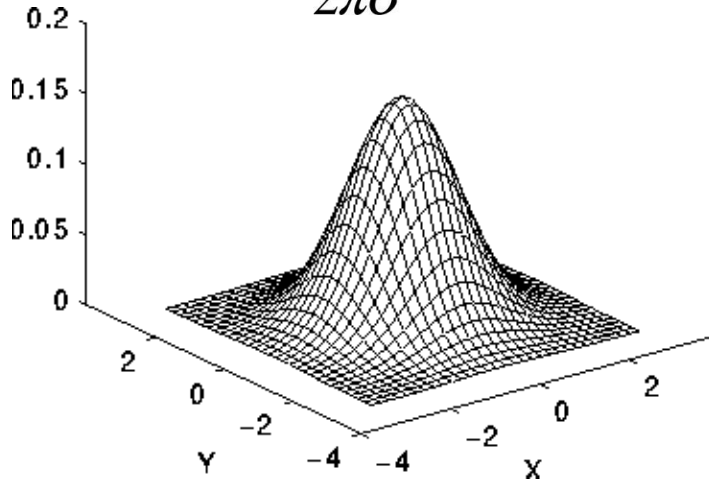
$\pi \approx 3.14159\dots$

$e \approx 2.71828\dots$

# Gaussian Smoothing

- ▶ Mask weights are samples of a Gaussian Function

$$G(x, y) = \frac{1}{2\pi\sigma^2} \exp\{-(x^2 + y^2)/2\sigma^2\}$$



$$\frac{1}{255}$$

1	4	6	4	1
4	16	26	16	4
6	26	43	26	6
4	16	26	16	4
1	4	6	4	1

5×5 Gaussian filter,  $\sigma=1$

# Gaussian Smoothing – RGB images

---



$$G(x, y) = \frac{1}{2\pi\sigma^2} \exp\{-(x^2 + y^2) / 2\sigma^2\}$$

---



# Gaussian Smoothing – animation

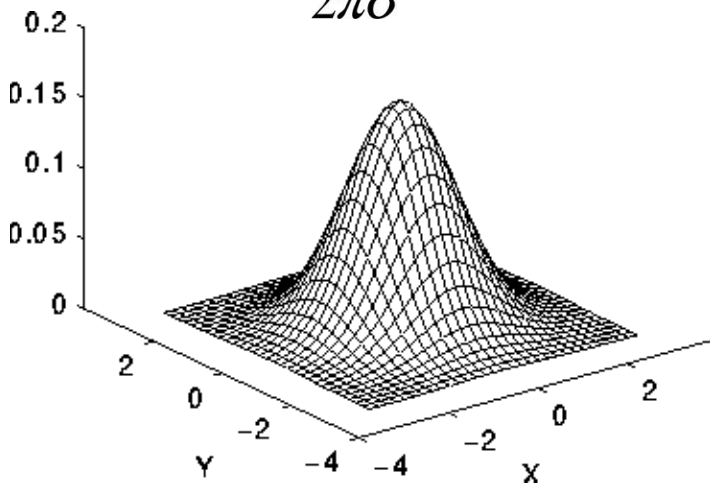
---



# Gaussian Smoothing

- ▶ Mask weights are samples of a Gaussian Function

$$G(x, y) = \frac{1}{2\pi\sigma^2} \exp\{-(x^2 + y^2)/2\sigma^2\}$$



$$\frac{1}{255}$$

1	4	6	4	1
4	16	26	16	4
6	26	43	26	6
4	16	26	16	4
1	4	6	4	1

5×5 Gaussian filter,  $\sigma=1$

- ▶ Relation between  $\sigma$  and smoothing?

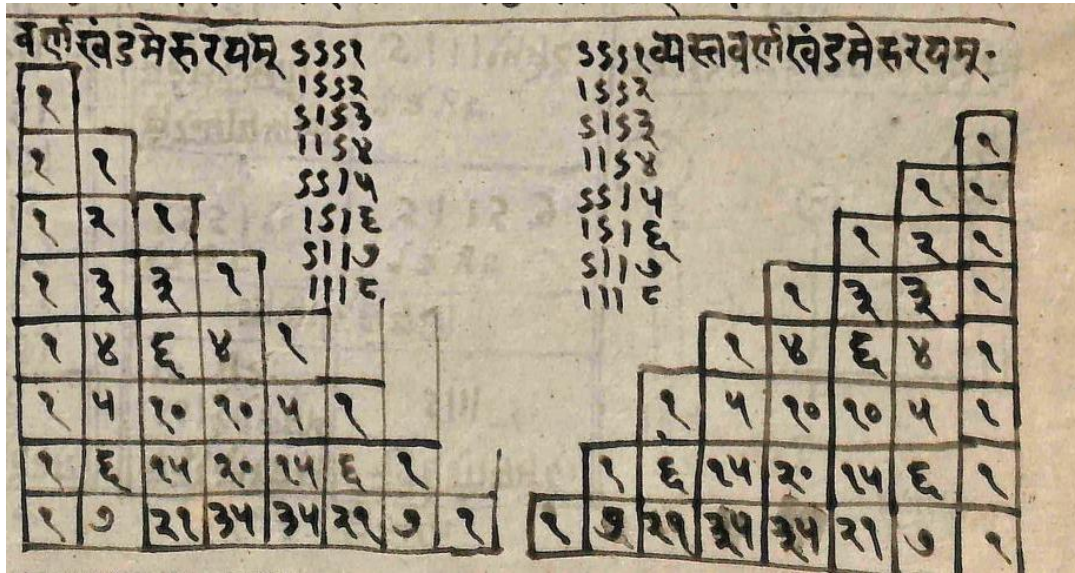


# Gaussian Smoothing – Effect of sigma



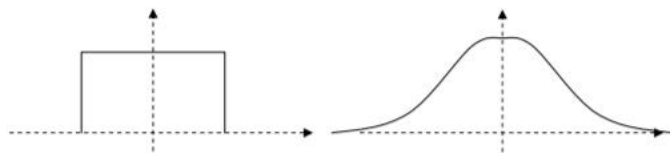
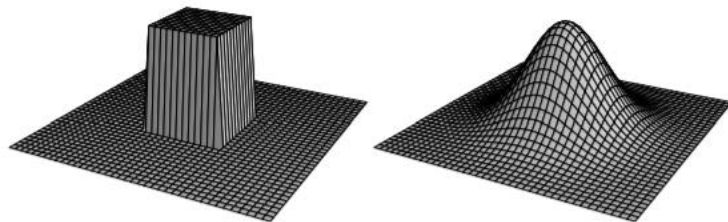
$$G(x, y) = \frac{1}{2\pi\sigma^2} \exp\{-(x^2 + y^2)/2\sigma^2\}$$

# Detour: Generating the coefficients



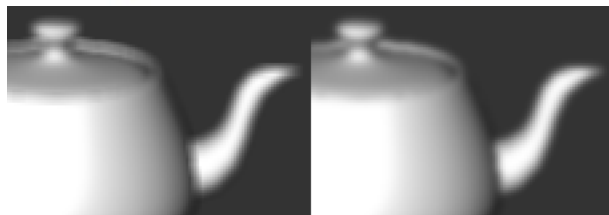
1	4	6	4	1
4	16	26	16	4
6	26	43	26	6
4	16	26	16	4
1	4	6	4	1

► Meru Prastaara, derived from Pingala's formulae (2 BCE), Manuscript from Raghunath Temple Library, Jammu



0	0	0	0	0
0	1	1	1	0
0	1	1	1	0
0	1	1	1	0
0	0	0	0	0

0	1	2	1	0
1	3	5	3	1
2	5	9	5	2
1	3	5	3	1
0	1	2	1	0

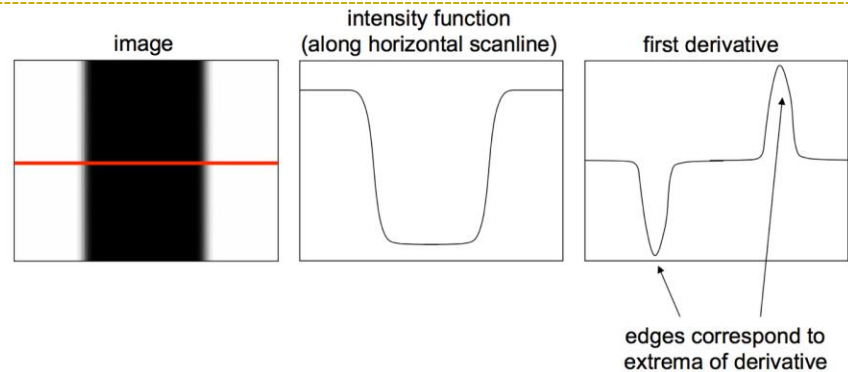


Out of focus blur



## ► First Derivative

$$\frac{\partial f}{\partial x} = f(x + 1) - f(x)$$

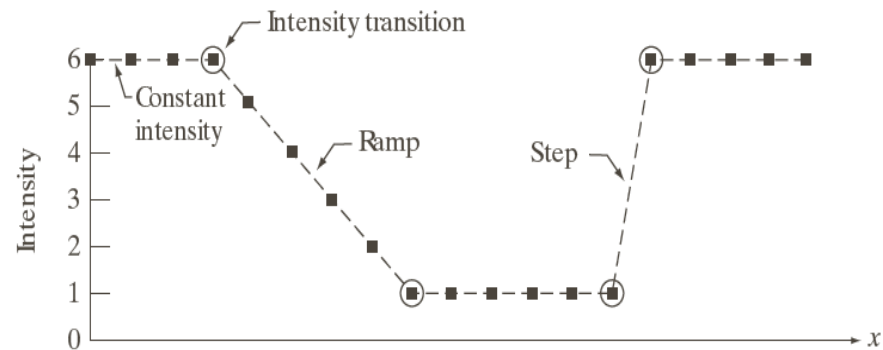


## First Derivative

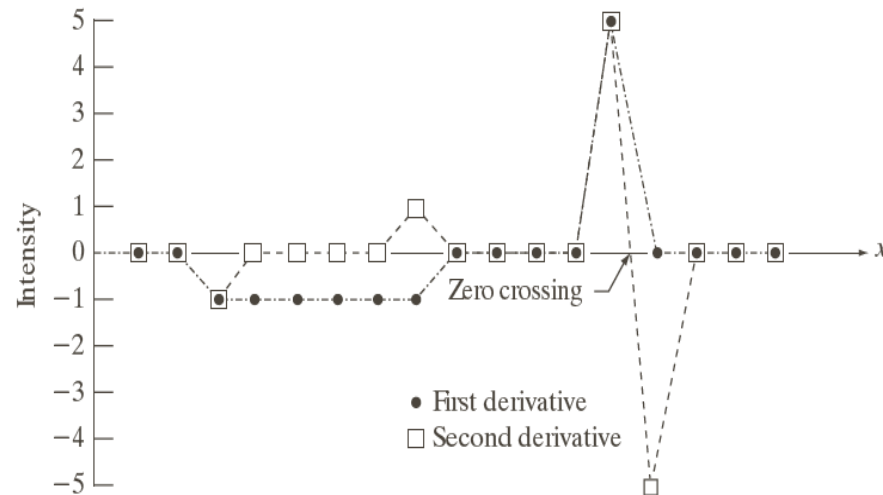
$$\frac{\partial f}{\partial x} = f(x+1) - f(x)$$

## Second Derivative

$$\frac{\partial^2 f}{\partial x^2} = f(x+1) + f(x-1) - 2f(x)$$



Scan line	6	6	6	6	5	4	3	2	1	1	1	1	1	1	6	6	6	6	6	x
1st derivative	0	0	-1	-1	-1	-1	-1	0	0	0	0	0	0	5	0	0	0	0	0	
2nd derivative	0	0	-1	0	0	0	0	1	0	0	0	0	0	5	-5	0	0	0	0	



# 1-D Derivatives

---

## ▶ First Derivative

$$\frac{\partial f}{\partial x} = f(x+1) - f(x)$$

- ▶ Zero in flat segments
- ▶ Nonzero at the onset of a step or ramp
- ▶ Nonzero along ramps

## ▶ Second Derivative

$$\frac{\partial^2 f}{\partial x^2} = f(x+1) + f(x-1) - 2f(x).$$

- ▶ Zero in flat areas;
- ▶ Nonzero at the onset and end of a gray-level step or ramp;
- ▶ Zero along ramps of constant slope



# Image Derivatives

---

$f(x-l, y)$	$f(x, y)$	$f(x+l, y)$
-------------	-----------	-------------

$$\frac{\partial^2 f}{\partial x^2} = f(x+1, y) + f(x-1, y) - 2f(x, y)$$

$f(x, y-l)$
$f(x, y)$
$f(x, y+l)$

$$\frac{\partial^2 f}{\partial y^2} = f(x, y+1) + f(x, y-1) - 2f(x, y)$$





# Laplacian Filter

---

$$\nabla^2 f = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

$$\nabla^2 f = f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) - 4f(x, y)$$

0	1	0
1	-4	1
0	1	0

0 response in 'flat' areas



# Laplacian Filter

0	1	0	1	1	1
1	-4	1	1	-8	1
0	1	0	1	1	1

0	-1	0	-1	-1	-1
-1	4	-1	-1	8	-1
0	-1	0	-1	-1	-1

a	b
c	d

**FIGURE 3.37**

(a) Filter mask used to implement Eq. (3.6-6).

(b) Mask used to implement an extension of this equation that includes the diagonal terms.

(c) and (d) Two other implementations of the Laplacian found frequently in practice.

# Sharpening Filter

---

- ▶ Objective of sharpening: Highlight fine detail / Enhance detail that has been blurred.
- ▶ Smoothing  $\rightarrow$  Averaging  $\rightarrow$  Summation
- ▶ Sharpening  $\rightarrow$  Difference



# Implementation

---

$$I'(u,v) = \begin{cases} I(u,v) - \nabla^2 I(u,v) \\ I(u,v) + \nabla^2 I(u,v) \end{cases}$$

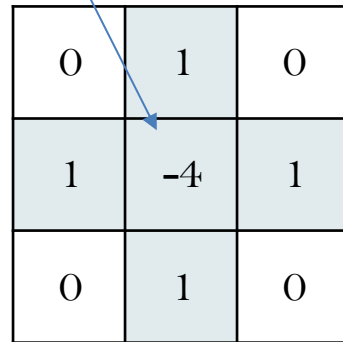
*If the center coefficient is negative*

*If the center coefficient is positive*

Where  $I(u,v)$  is the original image

$\nabla^2 I(u,v)$  is Laplacian filtered image

$I'(u,v)$  is the sharpened image



0	1	0
1	-4	1
0	1	0

*sharpened = range\_normalize(input + filtered )*

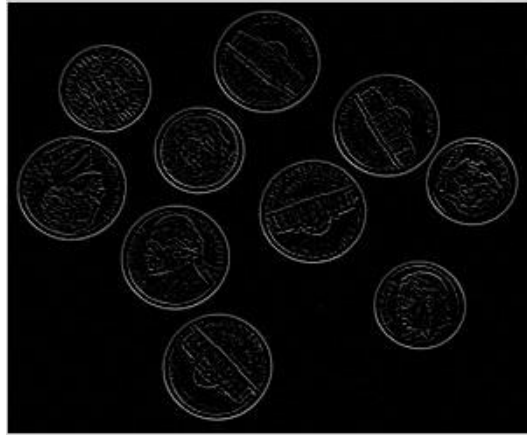
---

# Sharpening with Laplacian Filters

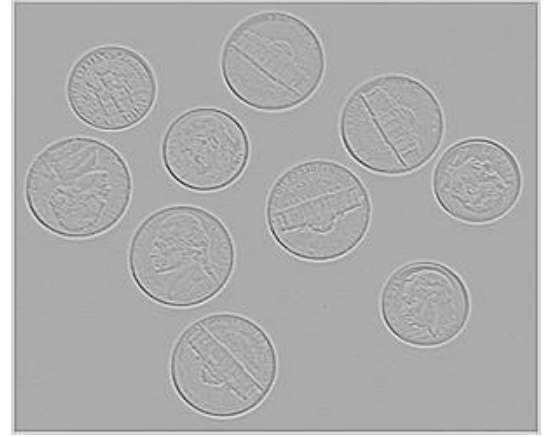
---



$$I(u, v)$$



$$\nabla^2 I(u, v)$$



$$\nabla^2 I(u, v) + 128$$

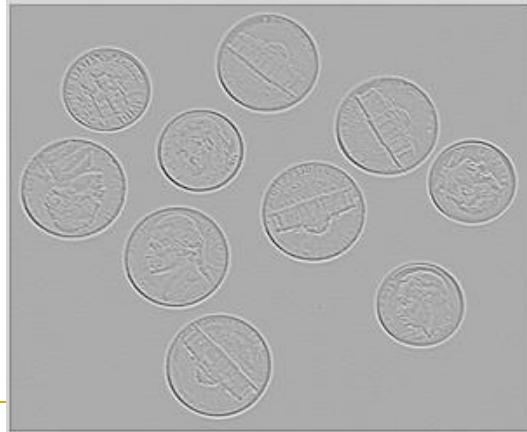
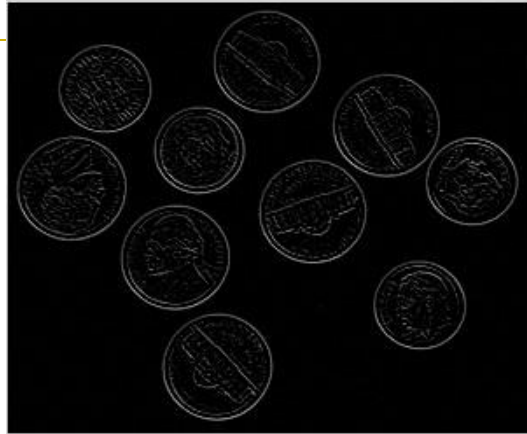
(For Visualization)

# Laplacian Filters $\nabla^2 I(u, v)$

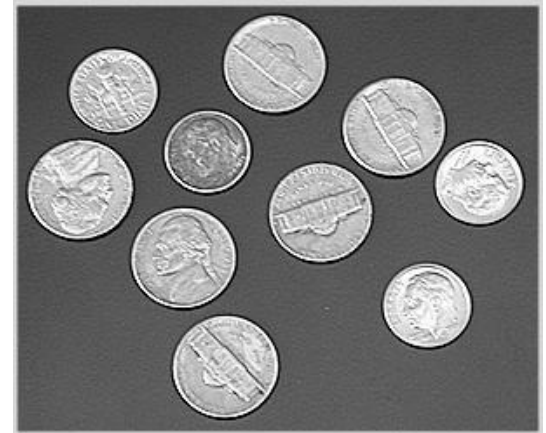
$I(u, v)$



$\nabla^2 I(u, v) + 128$   
(For Visualization)

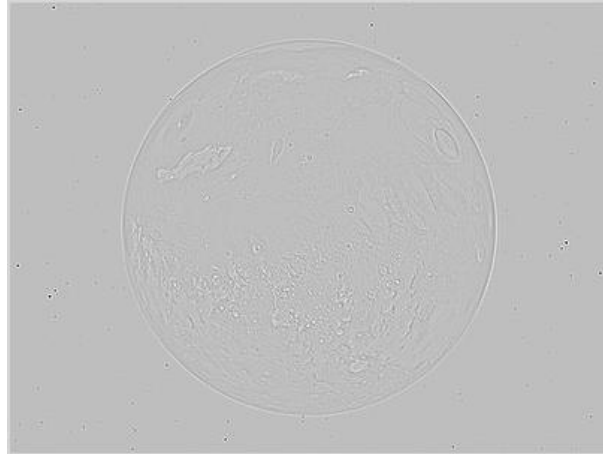


$I'(u, v)$



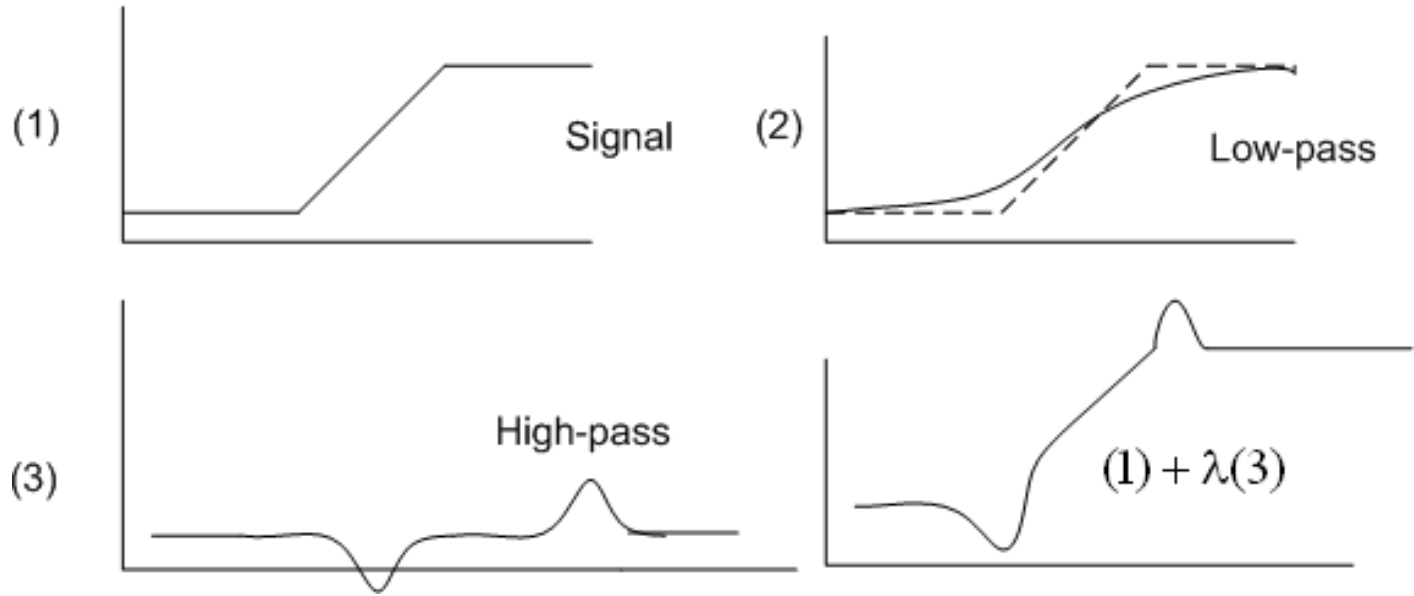
# Sharpening with Laplacian Filters

---



# Unsharp Masking (and Highboost Filtering)

---



$$g_{mask}(x, y) = f(x, y) - \bar{f}(x, y), \quad g(x, y) = f(x, y) + k * g_{mask}(x, y).$$





# Unsharp Masking (and Highboost Filtering)

---

- ▶ **High boost filter:** amplify input image, then subtract a lowpass image

$$\text{Highboost} = A \text{ Original} - \text{Lowpass}$$

$$= (A - 1) \text{ Original} + \text{Original} - \text{Lowpass}$$

$$= (A - 1) \text{ Original} + \text{Highpass}$$

(A-1)  +  = 

---



# Unsharp Masking / Highboost Filtering

---

$$A \geq 1$$
$$w = 9A - 1$$

-1	-1	-1
-1	w	-1
-1	-1	-1

$$A = 2$$
$$w = 17$$

-1	-1	-1
-1	17	-1
-1	-1	-1

- ▶ If **A=1**, we get unsharp masking.
- ▶ If **A>1**, part of the original image is added back to the high pass filtered image (highboost filtering).



# Unsharp Masking (and Highboost Filtering)



# Unsharp Masking (and Highboost Filtering)



**FIGURE 3.40**

- (a) Original image.
- (b) Result of blurring with a Gaussian filter.
- (c) Unsharp mask.
- (d) Result of using unsharp masking.
- (e) Result of using highboost filtering.

# Sobel Edge Masks

Original



Laplacian



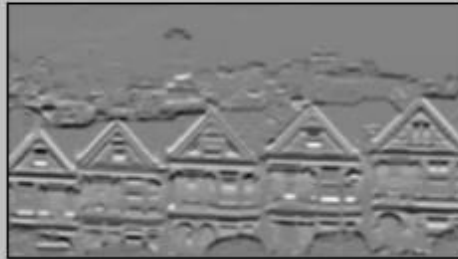
Sobel X

$$\begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix}$$



Sobel Y

$$\begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$



0	-1	0
-1	4	-1
0	-1	0

Original



Laplacian



Sobel X



Sobel Y



0	-1	0
-1	4	-1
0	-1	0

$$\begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix}$$

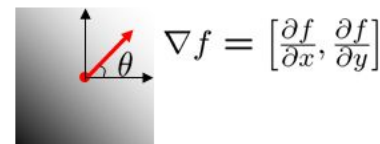
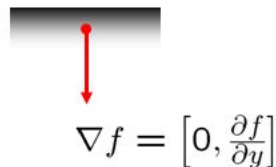
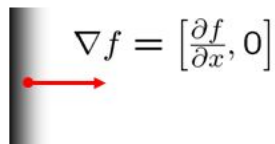
$$\begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

## Image gradient

The gradient of an image:

$$\nabla f = \left[ \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right]$$

The gradient points in the direction of most rapid change in intensity



The gradient direction is given by:

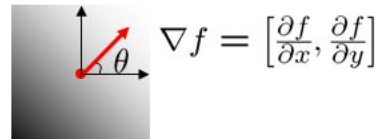
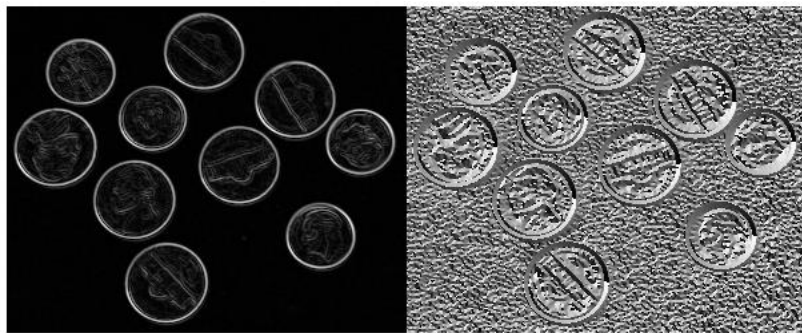
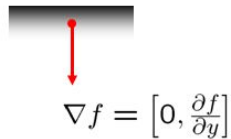
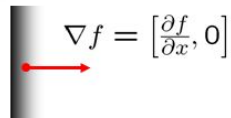
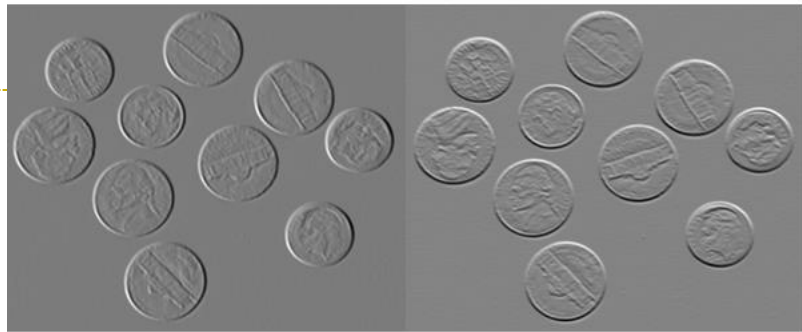
$$\theta = \tan^{-1} \left( \frac{\partial f}{\partial y} / \frac{\partial f}{\partial x} \right)$$

- how does this relate to the direction of the edge?

The *edge strength* is given by the gradient magnitude

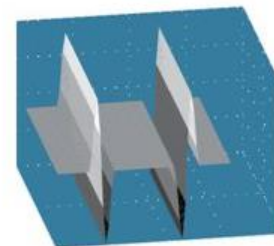
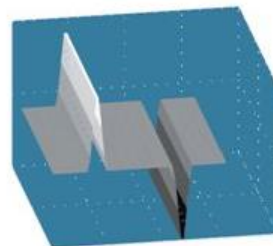
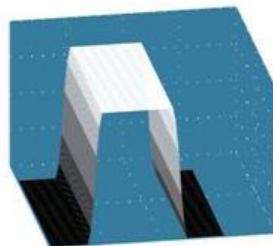
$$\|\nabla f\| = \sqrt{\left( \frac{\partial f}{\partial x} \right)^2 + \left( \frac{\partial f}{\partial y} \right)^2}$$

# Edge Magnitude and Gradient



$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2} \quad \theta = \tan^{-1} \left( \frac{\partial f}{\partial y} / \frac{\partial f}{\partial x} \right)$$



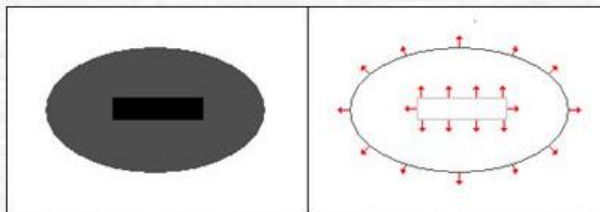


## Calculus...

- Rate of change in two directions
- Vector variable:
  - Gradient Magnitude
  - Orientation (0° is East)

$$\nabla f(i, j) = \sqrt{\left(\frac{\delta f(i, j)}{\delta i}\right)^2 + \left(\frac{\delta f(i, j)}{\delta j}\right)^2}$$

$$\phi(i, j) = \arctan\left(\frac{\delta f(i, j)}{\delta j}, \frac{\delta f(i, j)}{\delta i}\right)$$



# References

---

- ▶ GW Chapter – 3.4.1, 3.5.1, 3.6

